An Inefficient Representation of the Empty Word

Peter R.J. Asveld

Department of Computer Science, Twente University of Technology
P.O. Box 217, 7500 AE Enschede, the Netherlands
e-mail: infprja@cs.utwente.nl

1 Preface

From July 1978 till the summer of 1980 Paul Vitányi and I shared an office at Mathematical Centre (former name of C.W.I.), located at Tweede Boerhaavestraat 49 in the Oosterpark-neighborhood of Amsterdam. A couple of weeks after I arrived, Paul and I moved together with Arie de Bruin to a freshly painted room which happened to be the half of a former class room, like many offices in that old school building, separated from its companion by a rather thin wall. Without difficulty or any intention we could hear at least 50% of our neighbors' conversations as they could from ours. And the view from our room was something special too: a 19th century part of the former Amstel brewery which happened to be of too little interest to industrial archeology in order to survive in later days.

Another well-known feature of the old Mathematical Centre was its library: famous for its outstanding collection of books, journals and reports on mathematics and computer science, and remarkable for the dishes of rat poison on the floors, inviting the nightly intruders to commit suicide rather than nibbling at the precious volumes.

In 1978 both Paul and I finished our Ph.D. work on parallel rewriting, although our theses have very little in common; cf. [1, 7]. But we both had the intention to change our subject into the direction of computational complexity: a subject in which Paul achieved much more than I did; see e.g. [4], which also gives me the opportunity to turn to a more serious matter that is related to the title of this note.

2 Introduction

As we all know from [4], Kolmogorov complexity deals with short and efficient descriptions of objects, e.g., descriptions of strings over the alphabet $\{0, 1\}$. But in some cases there is a long, involved way to go before we arrive at the object to be described even in a context that looks anything but complex.

To be more precise, we will show in this note that it is possible to describe a very simple object (viz. the empty string) by means of a simple method (viz.

a well-known elementary instance of Post's tag system) in a very complicated way. To illustrate this point we need a few definitions and some notation.

3 Definitions, Examples and a Conjecture

Post's tag system is a rewriting system $T = (\Sigma, d, P, \omega_0)$ which consists of an alphabet Σ, a natural number d, a function $P : \Sigma \to \Sigma^\star$ and an initial string ω_0 over Σ. We obtain the string ω_{i+1} from ω_i —denoted by $\omega_i \Rightarrow \omega_{i+1}$— for $i = 0, 1, \ldots$ as follows. If ω_i equals the empty word λ, then $\omega_{i+1} = \lambda$. If σ is the first symbol of a nonempty word ω_i, then we append $P(\sigma)$ to the right end of ω_i and we delete the first d symbols from $\omega_i P(\sigma)$; this yields ω_{i+1}.

As an example —to which we will restrict our attention in the sequel— consider $T_0 = (\{0, 1\}, 3, P_0, \omega_0)$ with $P_0(0) = 00$ and $P_0(1) = 1101$. This instance T_0 has originally been introduced by Post in [5, 6], but there are still a lot of open questions in connection with T_0; cf. e.g. [8, 2, 3].

From a dynamical system point of view[1], there are three possibilities for the ultimate behavior of T_0 (as function of the initial string ω_0, of course); viz.
(i) T_0 explodes: for each natural n there is an ω_i such that its length exceeds n;
(ii) T_0 reaches a fixed point: T_0's only fixed point is the empty string λ;
(iii) T_0 enters a periodic cycle of which the period must be an even number. (And indeed, for each even natural number there exists an initial string that ends in such a cycle; cf. [3]).

As examples of (ii) we mention that each string ω_0 of the form 0^k reduces in k steps to λ: since $0^k \Rightarrow 0^{k-1}$, we have $0^k \Rightarrow^k \lambda$. Less trivially, $\omega_0 = 000100000$ yields in 13 steps λ because $000100000 \Rightarrow 10000000 \Rightarrow 000001101 \Rightarrow 00110100 \Rightarrow 1010000 \Rightarrow 00001101 \Rightarrow 0110100 \Rightarrow 010000 \Rightarrow 00000 \Rightarrow^5 \lambda$.

The case $\omega_0 = 100000100$ is an example of (iii); the (smallest) period is 6, since $\omega_{12} = \omega_{18} = \omega_{24} = \ldots$; cf. [2].

Since no examples of (i) are known, we have the following conjecture.

Conjecture 3.1. *For each initial string ω_0 over $\{0, 1\}$, the ultimate behavior of Post's tag system T_0 is either (ii) or (iii).* □

There are a few similarities between this conjecture and the famous $3n + 1$-problem[2]; see [3]. But apart from the validity of 3.1, it is clear that the only string that we can describe in terms of ultimate behavior of T_0 is T_0's single fixed point: the empty string λ.

4 The Result

The advance of modern computers enables us to simulate rewriting systems for quite a number of rewriting steps. A few years ago some people reported in the appropriate news groups of Internet that, according to their simulations,

[1] It is legitimate to consider T_0 as a non-linear discrete deterministic dynamical system; cf. [3] for an explanation.

[2] Also known as Ulam's problem, Collatz's problem, or Syracuse problem.

T_0 would probably explode for $\omega_0 = (100)^{110}$ or, equivalently, for $\omega_0 = 1^{330}$, thereby providing a possible counterexample to 3.1. Note that for the invalidity of 3.1 we need a proof and not a simulated initial part of a possibly infinite derivation.

But continuing the simulation to the very end shows a different picture: after 43 913 328 040 672 rewriting steps T_0 reaches the empty string. The maximal length of intermediate strings is 31 299 218; it occurs after 14 392 308 412 264 rewriting steps. Of course, the first 110 steps are trivial: for $i = 1, \ldots, 110$ we have $\omega_i = (100)^{110-i}(1101)^i$, whereas the end of the rewriting process looks like

1328040598 01110100001101110100000000001101000011011101
1328040599 10100001101110100000000001101000011011101110100
1328040600 000011011101000000000011010000110111010011101
1328040601 01101110100000000001101000011011101001101000
1328040602 01110100000000001101000011011101001101000000
1328040603 10100000000001101000011011101001101000000
1328040604 00000000001101000011011101001101000000001101
1328040605 0000000110100001101110100110100000011010000
1328040606 000011010000110111010011010000011010000
1328040607 01101000011011101001101000001101000000
1328040608 0100001101110100110100000110100000000
1328040609 000110111010011010000011010000000000
1328040610 11011101001101000001101000000000000
1328040611 11101001101000000110100000000000001101
1328040612 01001101000000110100000000000011011101
1328040613 0110100000011010000000000001101110100
1328040614 010000001101000000000000110111010000
1328040615 000001101000000000000011011101000000
1328040616 001101000000000000110111010000000000
1328040617 10100000000000110111010000000000
1328040618 000000000000110111010000000001101
1328040619 00000000011011101000000000110100
1328040620 0000001101110100000000011010000
1328040621 000110111010000000001101000000
1328040622 11011101000000000110100000000
1328040623 1110100000000001101000000001101
1328040624 0100000000001101000000011011101
1328040625 000000000110100000001101110100
1328040626 00000011010000000110111010000
1328040627 0001101000000001101110100000
1328040628 1101000000001101110100000000
1328040629 1000000001101110100000001101
1328040630 0000001101110100000011011101
1328040631 000110111010000000101110100
1328040632 11011101000000010111010000
1328040633 1110100000000101110100001101

1328040634 01000000001101110100001101110 1
1328040635 00000011011101000011011101 00
1328040636 00001101110100001101110100 00
1328040637 01101110100001101110100 0000
1328040638 01110100001101110100000000
1328040639 10100001101110100000000
1328040640 00001101110100000000001101
1328040641 01101110100000000000110100
1328040642 01110100000000000110100 00
1328040643 10100000000001101000000
1328040644 00000000001101000001101
1328040645 0000000110100000110100
1328040646 000011010000011010000
1328040647 01101000001101000000
1328040648 0100000110100000000
1328040649 000011010000000000
1328040650 001101000000000000
1328040651 10100000000000000
1328040652 000000000000001101
1328040653 00000000000110100
1328040654 0000000011010000
1328040655 000001101000000
1328040656 00110100000000
1328040657 1010000000000
1328040658 00000000001101
1328040659 0000000110100
1328040660 000011010000
1328040661 01101000000
1328040662 0100000000
1328040663 000000000
1328040664 00000000
1328040665 0000000
1328040666 000000
1328040667 00000
1328040668 0000
1328040669 000
1328040670 00
1328040671 0
1328040672 λ

of which the last 9 steps are again trivial. The first column in this table shows $i \bmod (2 \cdot 10^9)$. An overview of the string length as function of the discrete time parameter i is depicted in Figure 1.

The simulation of T_0 for $\omega_0 = (100)^{110}$ is no simple task; it takes about 80 MIPSyears. Clearly, this number depends on the way the simulation is coded but, as far as I can see, there is not much to be gained. Note that, due to

the sequential character of the rewriting process, parallelism is of no use either. Storing all intermediate strings is out of the question, since this would take about $214 \cdot 10^9$ Gb. And still you want to store some useful information for making snapshots like Figure 1, and for the unlucky moments that your system crashes.

5 Conclusion

We showed that it is possible to describe the empty string in a very inefficient way using rather simple means only, and that Conjecture 3.1 stands firm as ever.

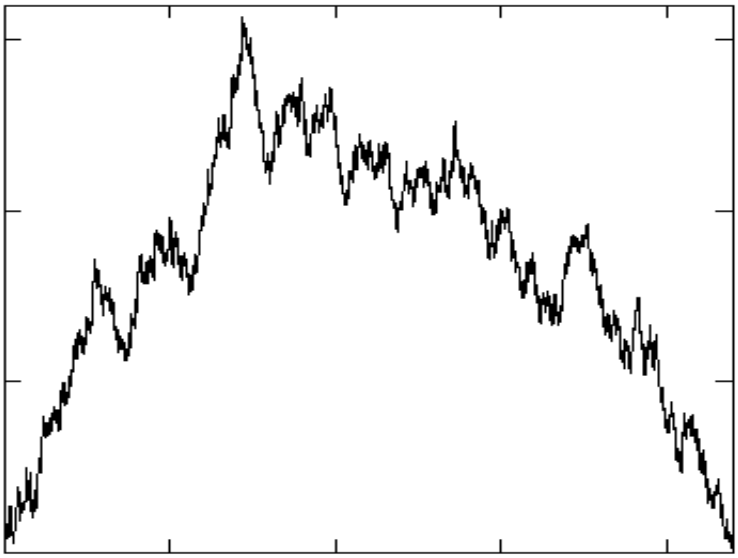

Figure 1.

References

[1] Asveld, P.R.J.: *Iterated Context-Independent Rewriting – An Algebraic Approach to Families of Languages* (1978), Twente University of Technology, Enschede.

[2] Asveld, P.R.J.: On a Post's system of tag, *Bull. Europ. Assoc. Theor. Comput. Sci.* No. 36 (1988) 96–102.

[3] Asveld, P.R.J.: Post's system of tag – A simple discrete nonlinear system, pp. 26–31 in: J.P. van der Weele (ed.): *Proceedings of Nonlinear Dynamics – Twente 91* (1992), Center for Theoretical Physics, University of Twente, Enschede, The Netherlands.

[4] Li, M. & Vitányi, P.: *An Introduction to Kolmogorov Complexity and Its Applications* (1993), Springer-Verlag, New York, Berlin, Heidelberg, etc.

[5] Post, E.M.: Formal reductions of the general combinatorial decision problem, *Amer. J. Math.* **65** (1943) 197–215.

[6] Post, E.M.: Absolutely unsolvable problems and relatively undecidable propositions — An account of an anticipation, pp. 340–433 in: M. Davis (ed.): *The Undecidable — Basic Papers on Undecidable Propositions, Unsolvable Problems and Computable Functions* (1965) Raven, New York.

[7] Vitányi, P.M.B.: *Lindenmayer Systems – Structure, Languages, and Growth Functions* (1978), Free University, Amsterdam.

[8] Wanatabe, S.: Periodicity of Post's normal process of tag, pp. 83–99 in: *Proc. Symp. on Math. Theory of Automata 1962* (1963), Polytechnic Press, Brooklyn, N.Y.